Examining the Structure of Venture Capital Investment Networks

Kevin Wu
kwu93@stanford.edu

Timothy Lee
timothyl@stanford.edu

Anthony Ma
akma327@stanford.edu

***Abstract*— Our paper focuses on examining the structure of venture capital investment networks. In examining investment networks, we construct two types of networks. First is a bipartite investment network G with unweighted directed edges from investors to companies based on investment made, and a more complicated formulation with weights equivalent to funding amount. The second is a folded coinvestment network F on only investor nodes with undirected edges between investors such that an edge $(u, v) \in F$ if and only if $(u, c) \in G$ and $(v, c) \in G$; that is, two investors u and v made an investment in the same company c. One part of our research focus was to understand the structure of these networks and their properties such as degree distribution. The second focus aims to tackle the link prediction task on both the folded coinvestment network F, and the bipartite investment network G, where the latter is assumed to be a much more difficult task. Using a logistic regression and SVM model with walk-forward cross validation on our time-partitioned graphs, we see decent but not time-robust predictive accuracy on link prediction in the folded coinvestment network F; however, the results of link prediction on the actual bipartite investment graph G leave much to be desired. We conclude that link prediction, particularly in private investment networks, is incredibly difficult given a surface-level dataset, and limited graph features of Bonacich centrality, Adamic-Adar, and Weighted-Katz metrics.**

I. INTRODUCTION

Venture capital (VC) is a very important domain in finance that helps nascent companies raise capital to see their ideas to fruition. Identifying investment opportunities in VC, however, is exceptionally difficult because of relative scarcity of data in terms of observable information and outcomes compared to those in public markets. However, because of technological advancements and the democratization of information through services like Crunchbase or Mattermark, ordinary individuals can now attain a higher degree of visibility on the entire VC investing landscape.

In particular, one can now build out a relatively large network filled with VC firms and startups to examine the structure of VC investing in a bipartite investment network. Given this data, our project aims do this by identifying certain structures in the network and measure characteristics such as centrality and identifying relevant features that might be useful for link prediction of a VC firm investing in a particular startup.

II. PREVIOUS LITERATURE

Nowell and Kleinberg [3] proposed several methods to predict links between nodes based on pre-existing edges in a graph and empirically tested these methods on various co-authorship networks. One of the methods used in this paper leverages Adamic-Adar and Katz metrics which we will also use in our analysis. Their work is particularly enlightening for our project not only due to its broad survey of the many methods of link prediction but also the analogous relationship between co-authorship of papers and co-investment of companies. However, for our specific problem of co-investment of companies, it is also useful to consider the nature of the investment (round of funding and amount raised) and different features of the investors and company (e.g. location and market) in approaching the link prediction problem.

Bellavitis et. al. [1] examines the tradeoff between cohesiveness of venture capital (VC) investment networks versus those that are rich in structural holes. When a VC firm makes a decision to co-invest with partners who are already in its network, they increase the cohesiveness of their network. On the other hand, when a VC firm decides to establish new ties, it increases the structural holes in the network. We use this idea as a motivation for understanding the connectivity of an investment network and as a motivation for folding a network onto itself in an attempt to express the connectivity among investors.

Finally, Liang and Yuan [2] propose a novel perspective in predicting the outcomes of investment links. Many prior studies have been done on investment behaviors in terms of personal opinions, investment experience, and geography. However, the authors here argue that social dynamics between investors and companies are actually much more telling of eventual investment outcomes. Several machine learning algorithms were performed upon a wide variety of features that represent the social dynamics between stakeholders. However, it would be interesting to consider additional features such as sentiment, reputation, novelty of a company, and other social factors that cannot be simply inferred from network graph structure. We can apply these features to optimize supervised machine learning algorithms to enhance accuracy for investment link prediction.

III. METHODS

Our main goal is to perform link prediction in a bipartite venture capital investment graph and a folded coinvestment network. Prior to this, we explore the data and understand characteristics of our network and use descriptive statistics to characterize aspects of our network. This will allow us to identify if there exist any interesting signals within the network we can use to generate features for link prediction. These include various forms of centrality, as well as measures for network cohesion. After defining features, we split our dataset into training and test sets by time and determine the accuracy of link prediction on our test set.

A. Data Collection

For this project, our data source is Crunchbase [4], which is a public database that consists of information about startup companies and investors. We retrieved several CSV files from Crunchbase which allow us to analyze data on 1) Companies and 2) Investments. Specifically, we have data on roughly 47758 companies, 110044 different investment transactions, and 21350 unique investors. For companies we have information on what industry they operate in, what series of funding they are on at a particular date, as well as total funding raised. The Investments dataset includes information about the transaction between investor and company, in what fiscal quarter it took place, and markets that the investor and company operate in.

B. Link Prediction

Next, we will define our main problem: prediction of links. There are two problems at hand: First, we will be predicting for a historical network of coinvestments among investors, which investor-investor pairs are likely to be co-investors in some company in the future. Specifically, we take the coinvestment graph at a certain time-step, extract and learn predictive features of the graph, and use the additional edges created after

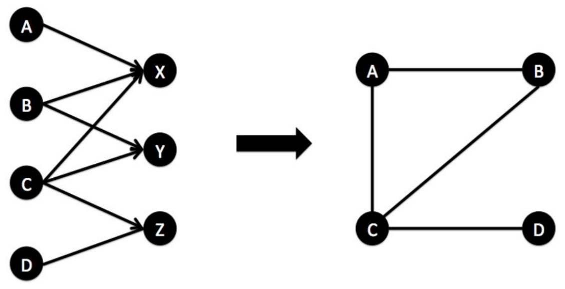


Fig. 1: The networks shown above illustrates the result of folding the bipartite network on the left onto the left cluster, generating the folded network on the right. In our problem, we consider nodes A, B, C, and D to be investor nodes and X, Y, and Z to be company nodes.

the time-step as a test set of predicted edges. Some of these predictive features, as introduced in Lowell and Kleinberg, include the metrics Adamic-Adar and Weighted Katz and are further described in the Features section. The second problem is prediction of future investments from an investor to company. Given a historical coinvestment graph, historical investment graph, and contextual variables about companies and investors, this is a harder problem than the first since two investors have many more chances to coinvest in a company, given the large number of companies available, than the situation where we have one specific investor and one specific company. Thus, we hypothesize the features and model for this problem will be much more complex, and we will ultimately relax the constraints of the problem to make it more feasible to predict links.

IV. GRAPH REPRESENTATIONS

In examining investment networks, we construct two types of networks. First is a bipartite investment network G with unweighted directed edges from investors to companies based on investment made, and a more complicated formulation with weights equivalent to funding amount. Figure 2 shows a small section of the investment network we are working with, limited to the top investors and companies. The second is a folded coinvestment network F on only investor nodes with undirected edges between investors such that an edge $(u, v) \in F$ if and only if $(u, c) \in G$ and $(v, c) \in G$; that is, two investors u and v made an investment in the same company c. In addition, for the coinvestment network, we enforce the condition that the two investors must have invested in the same company within the same fiscal year to more accurately represent the notion of a coinvestment. The coinvestment network is more lossy than the bipartite network because the bipartite network has edges between companies and investors that do not correspond to co-investment edges. Figure 1 shows a pictorial example of how a bipartite network is folded onto itself.

To capture more information within each edge in each kind of network, we add an edge weight that describes the amount of stake an investor has in a company. For the bipartite network G, we define the weight of an edge (u, c) to be $\log_{10} raisedFunds(u, c)$ where $raisedFunds(u, c)$ amount in US dollars that u raised for company c. On the other hand, for the folded network F, we define an edge weight that describes the amount of coinvestment stake the two investors had in a company, defined as $\log_{10} raisedFunds(u, c) + \log_{10} raisedFunds(v, c)$ where $raisedFunds(u, c)$ and $raisedFunds(v, c)$ are the amount in US dollars that u and v respectively raised for company c.

Since our data can be modeled as a time series, with new investments happening across time, we construct different graphs based on the investments made in a particular year t. That is, G_t represents the investments network with edges that correspond to investments made only in year t. F_t represents the coinvestment network with only edges between investors that represent coinvestments made only in year t.

V. Features

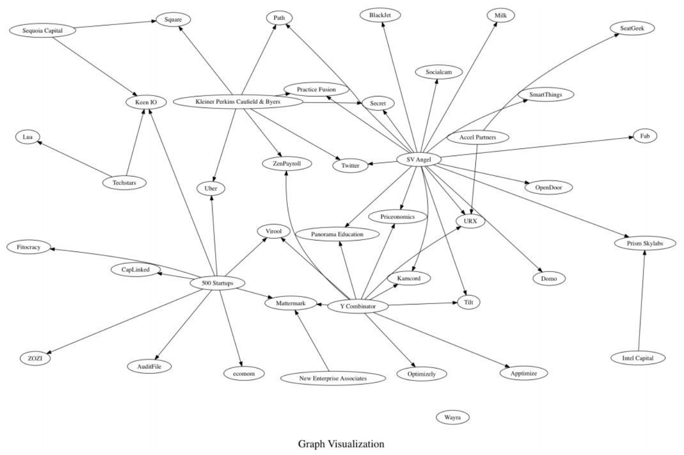
In this section, we describe several measures that we rely on for both the bipartite link prediction problem and the folded network link prediction problem. We use features extracted from the networks F and G and in addition, descriptive variables about the individual company and investor, independent of the network.

A. Adamic-Adar

$$A(x, y) = \sum_{z \in \Gamma(x) \cap \Gamma(y)} \frac{1}{\log |\Gamma(z)|}$$

Between nodes x and y, z represents the set of nodes found in both sets of neighbors, $\Gamma(x)$ and $\Gamma(y)$, and each neighbor is weighted by the inverse of its relative frequency, or $\frac{1}{\log |\Gamma(z)|}$ (i.e. rarer nodes are more indicative of similarity). This metric intuitively captures the connectedness between two nodes based on the number of shared neighbors.

Fig. 2: Induced Subgraph of Top Venture Firms and their Company Investments



This graph represents the top 10 investors across time (excluding ”Start-Up Chile”) and the companies they have invested in, limited to those that have at raised money from at least 25 investors (for visualization purposes). This represents the top investors and the ”popular” companies they invest in. As we can see, these are mostly all notable entities that one would associate with the field of venture capital.

B. Weighted Katz

$$K(x, y) = \sum_{l=1}^{\infty} \beta^l * |paths_{x,y}^{<l>}|$$

Between nodes x and y, $paths_{x,y}^{<l>}$ represents the set of all length-l paths between the two nodes. β is a learned parameter between 0 and 1 and weights closer paths more when it is low. This feature intuitively represents the degree of connectedness between two nodes, and is different from the Adamic-Adar metric in that it allows for two nodes to be connected in paths that more than just two steps away.

C. Bonacich Centrality

$$c_i(\alpha, \beta) = \sum_j (\alpha - \beta c_j) R_{i,j}$$

This measurement of centrality differs from other forms of centrality in that it can take into account the centrality of a node's neighbors. It incorporates a family of centrality measures depending on the chosen value of β. $R_{i,j}$ denotes the presence of an edge between nodes i and j. In the case where β is 0, the metric is the same as degree centrality. However, in the case that the

β parameter is positive, anyone who knows influential people are made more influential themselves. When β is negative, being connected to people of low influence increases one's own status. The centrality measure for every node is given as follows and can be solved by power iteration.

With this feature, we can understand whether VC firm status in addition to the Adamic-Amar and Weighted Katz features would improve a model, and to what extent the difference in status between two VC firms would affect co-investment in the future. Specifically we add to the feature vector of a pair of investor nodes the two Bonacich centralities of the investor nodes, and the absolute difference of their centralities (representing the difference in status).

D. Contextual Indicator Variables

We construct additional features using contextual knowledge about how investors invest in companies more preferentially in terms of market preference or geographic proximity. That is, for a period of several years, there tends to be strong autocorrelation in terms of investments made in companies of a particular market. This motivates using the market that the company is in as a feature, as being in a "hot" market may increase the likelihood of an edge existing between a company and an investor. We define a "hot market" as a market that is in the top 20 list of markets with the highest number of investments. Figure 3 shows some of the top markets by investment count. We can encode this feature as the alignment of a company being in a hot market, and the investor having concentrated at least 15% of their investments during the training period in companies of that particular market.

$$\begin{aligned}&\text{GoodMatch}(u, c, F) := \\ &I\left[\text{market}(c) \in \text{HotMarkets}(F)\right] \times \\ &I\left[\text{market}(c) \in \text{InterestedMarkets}(u)\right]\end{aligned}$$

These features are generated only for the prediction of links in the bipartite network. It allows for the chance that even if not a single investor has yet invested in the company, an investor may still consider investing in the company if it is interested in the same market as the company exists in.

E. Sum of Weighted Edges

$$sumWeights(u, v) = \sum_{\substack{j \in Edges(F) \\ j_1 = u \\ j_2 = v}} w_j$$

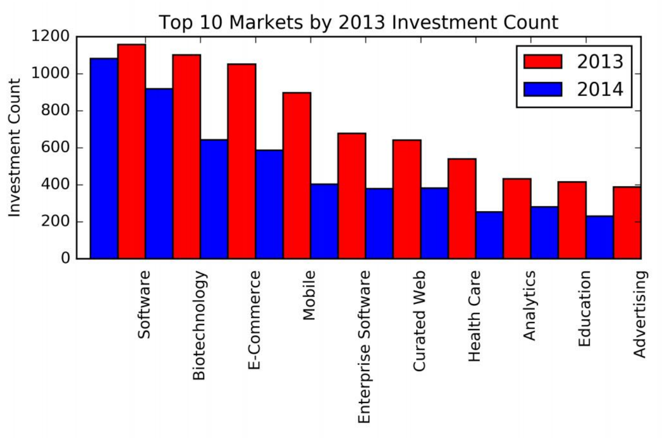


Fig. 3: Examining the number of investments grouped by the market that the company exists in between 2013 and 2014, we see a year on year autocorrelation. The year over year autocorrelation between 2013 and 2014 95.6% when examining the top 30 markets in each year, indicating that company market may be an interesting feature for use in link prediction.

Given a pair of investors (u, v) in the network F, we use the sum of the weighted edges between the two investors to represent the total amount of co-investment between two investors across multiple companies in the past. The weight of each edge w_j is again, $\log_{10} raisedFunds(u, c) + \log_{10} raisedFunds(v, c)$ where $raisedFunds(u, c)$ and $raisedFunds(v, c)$ are the amount in US dollars that u and v respectively raised for company c. This feature is used only for the prediction of links in the coinvestment network.

VI. Models

In this section, we describe models that can be used to learn how to use the above features to predict whether a link will exist in the future.

A. Logistic Regression

$$h_\theta(x) = g(\theta^T x) = \frac{1}{1 + e^{-\theta^T x}}$$

The logistic regression model fits a logistic function to a set of feature vectors. Each feature vector x, has a score z that is equal to $\theta^T x$ where theta corresponds to a vector of weights, one for each feature. For a given score, z, you obtain from the logistic function a certain probability between 0 and 1 that the class is true. The weights can be obtained by finding the weights that maximizing the likelihood of the training

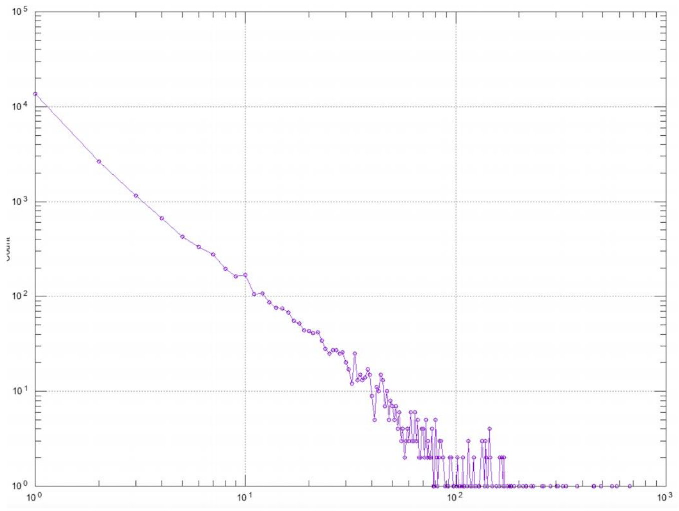


Fig. 4: The degree distribution of the bipartite investment network (outdegree for investors, indegree for companies). We see that the degree distribution follows similar to the Preferential Attachment Model.

examples, done through gradient ascent or Fisher scoring. We use this logistic regression model and different forms of regularization techniques such as specifying a regularization penalty $C = \frac{1}{\lambda}$ where higher λ terms more heavily penalize larger model coefficients.

B. Support Vector Machines

$$HingeLoss(x, y|\theta) = max(1, 1 - (\theta^T x)y)$$

The support vector machine model approaches binary classification by finding a separating hyperplane that separates the two classes of training examples with the largest margin. This separating hyperplane is a linear model and the corresponding score, z = $\theta^T x$, if greater than zero, denotes a positive classification. Finding the weights of the linear model corresponds to minimizing the hinge loss over the training descent. Stochastic gradient descent which minimizes the hinge loss over a single training example in a single iteration also finds the weights that minimize the hinge loss.

C. Preferential Attachment Model

Examining the degree distribution of the bipartite investment network indicates that the distribution is similar to that of the preferential attachment model.

We consider the link prediction problem in the context of the preferential attachment model; probabilistically predict edges based on degrees, i.e., the preferential attachment score $z(x, y)$ between an investor node x and a company node y described as

$$z(x, y) = \frac{1}{C}|\Gamma(x)||\Gamma(y)|$$

being greater than some parameter γ (where C is a normalization constant). However, previous literature [5] has shown that this model doesn't work very well and is also not deterministic, so we focus mainly on feature construction and usage of Logistic Regression and SVM models.

VII. ALGORITHMS

We introduce two algorithms: one on the folded network and the second on the bipartite investment network.

A. Link Prediction on a Folded Network

In the link prediction problem over the coinvestment network, we predict for two investor nodes in a historical coinvestment graph $F_{tStart:tEnd}$, where we consider coinvestments between the years $tStart$ and $tEnd$, whether they will have another coinvestment in the year $tEnd + 1$. To train a model, we pick a set of pairs of investor nodes, 400 of which have a coinvestment in the year $tEnd$, and 400 of which do not. Note that since we have a 50/50 split in the labels, and that the coinvestment graph is generally not complete, we have severely downsampled the number of non-edges in the graph. For each pair, we generate the above numerical features described in the Features section from the coinvestment graph $F_{tStart:tEnd-1}$ and the label each pair with whether there was a corresponding coinvestment in year $tEnd$. We fit the features and response variable into both SVM and logistic regression models. Next, we use this model to predict coinvestments of the year $tEnd + 1$ in a set of pairs of investor nodes, with their corresponding features generated on the coinvestment graph $F_{tStart:tEnd}$. We can evaluate our model based on its predicted labels by cross-referencing it against the coinvestment network in the year $tEnd + 1$.

B. Link Prediction on a Bipartite Investment Network

In the link prediction problem over the bipartite investment network, we predict for one investor node and one company node in a historical investment graph $G_{tStart:tEnd}$, where we consider investments between the years $tStart$ and $tEnd$, whether the investor will invest in the company in the year $tEnd+1$. In addition to information from network $G_{tStart:tEnd}$, we will use information from the coinvestment graph of the same time period, $F_{tStart:tEnd}$. Define U as the set of investor nodes and C as the set of company nodes. We generate our features and response by iterating over all investor-company pairs of (u, c) for $u \in U$ and $c \in C$, *regardless of whether investor* u *actually invested in company* c. If $(u, c) \in G_{tStart:tEnd}$, then we add a 1 label to our response vector, indicating an edge, otherwise a 0 label.

Now we need to generate features for every (u, c) pair. We do this by taking the sum feature metric for $(u, u') \in F_{tStart:tEnd}$ for all $(u', c) \in G$. That is, considering the Adamic-Adar metric, given an investor u, for all other investors u' that are investing in company c, we compute the sum of the Adamic-Adar metrics for (u, u') that was generated from our folded coinvestment network and assign this as the feature value for (u, c). We use the sum to capture the additive effect of having multiple similar investors investing in that particular node. This describes the construction of our response label vector, and our feature vector for the training period. We fit a model to this data and then use it to predict that in the test period. Finally, we down-sample the number of non-edges in order to make the prediction task feasible. We do this since there are multiples more non-edges than edges and so we down-sample the training set and test set so that they contain equal numbers of samples of edges and non edges.

C. Walk Forward Cross-Validation

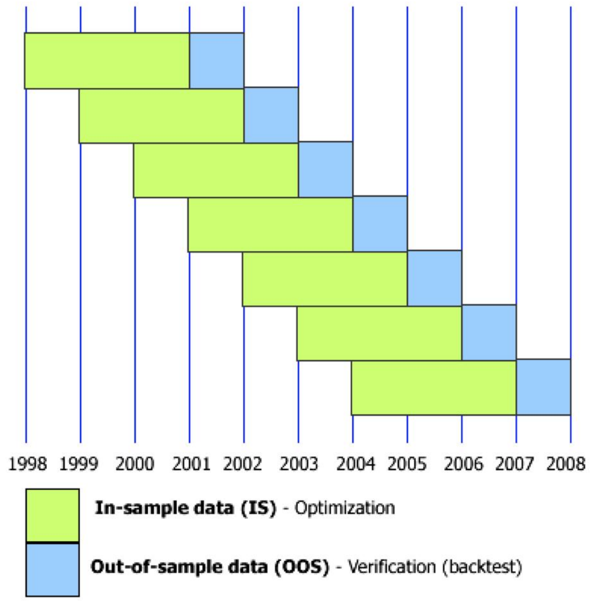
We evaluate the performance of our models by training on our model on the features generated from a subgraph in a particular year, and then testing the model on the following year. However, this exposes the model to certain biases when the distribution of features is dissimilar between training and test periods, or an abnormality arises in the training or test period. To evaluate the performance of our models more robustly with respect to time-variations, we introduce walk-forward cross-validation on our time-series data. Define our time series data as D; further, we define D'_t as the data collected in a particular year t', and $D_{t':t''}$ as data collected from years t' to t''. For the parameters into the algorithm, define a training window of T and a start period t_0 and number of iterations k. Call our model M. Then our walk forward optimization works as described in the Algorithm 1 pseudocode.

Algorithm 1: EvaluateModelCV(M, D, T, t_0, k)

```
initialize errors_list;
for iteration $i$ from $0 \to k-1$ do
    trainStart $\leftarrow t_0 + i$;
    trainEnd $\leftarrow$ trainStart $+ T$;
    trainGraph $\leftarrow D_{\text{trainStart:trainEnd}}$;
    testGraph $\leftarrow D_{\text{trainEnd+1}}$;
    X_train , Y_train $\leftarrow$ constructFeaturesAndResponse(trainGraph);

    X_test , Y_test $\leftarrow$ constructFeaturesAndResponse(testGraph);

    M.fit(X_train,y_train);
    error $\leftarrow$ compare(y_test, $M$.predict(X_test));
    add error to errors_list;
output mean error of errors_list;
```



1998
1999
2000
2001
2002
2003
2004
2005
2006
2007
2008
In-sample data (IS) - Optimization
Out-of-sample data (OOS) - Verification (backtest)

Fig. 5: Visualization of walk forward cross validation methodology. This allows us to evaluate our model in a way that is more robust to time inconsistencies and anomalies in our training (in-sample) and test (out-of-sample) data over a longer period of time. Additionally, this methodology is better for time-series data than traditional k-fold cross validation since we are only moving forward in time to ensure point-in-time consistencies are more strictly enforced.

VIII. RESULTS AND ANALYSIS

A. Coinvestment Link Prediction

Table I shows the accuracies on the training, cross-validation, and test set after fitting a model to predict

Model	Training Error	Cross Validation Error k=10	Test Error
SVM No Bonacich	0.135	0.141	0.147
Logistic Regression No Bonanacich	0.135	0.143	0.145
SVM With Bonacich	0.170	0.170	0.178
Logistic Regression With Bonacich	0.135	0.148	0.155

TABLE I: Folded Coinvestment Link Prediction Results: This table describes the training error and cross-validation error after training our model with either logistic regression or SVM. We depict whether the model included the Bonacich Centrality measures as features in the model with the name, "No Bonacich" and "With Bonacich" respectively. The features of the training data were calculated from the coinvestment graph, $F_{2005:2012}$ and labels from F_{2013}. The features of the test data were calculated from $F_{2005:2013}$ and labels from F_{2014}.

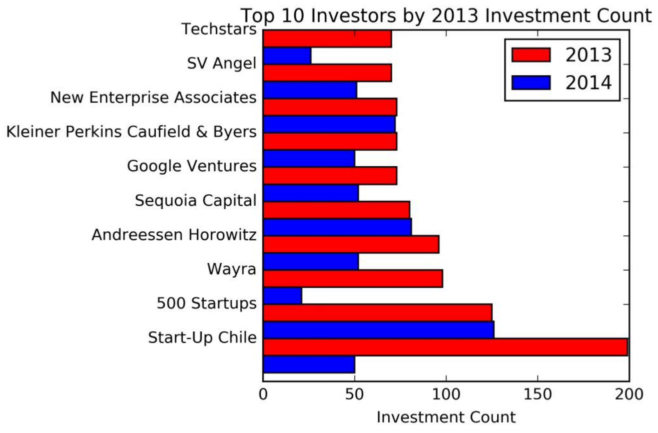


Fig. 6: As we examine the top investors by investment count, we see why it's important to use the similar set of investors in the training and test period: there is high variance in the investment counts among the top investors in 2013 and 2014.

coinvestments of the year 2014. We see that the logistic regression classifier worked slightly better than the SVM classifier, and that the information about the Bonacich centralities of the two investors surprisingly added little to the accuracy of the classifiers. We deem that our model generated has reasonable performance during cross-validation and on the training and test sets.

One hypothesis why the Bonacich centrality measures added little information to the model can be that since the coinvestment graph follows a preferential attachment model as shown in Figure 4, the graph tends to form separate clusters of high degree nodes surrounded by lower degree nodes. These higher degree nodes generally have higher Bonacich centrality measures. As a result, if the model attempts to predict a co-investment between two popular investors based on Bonacich centrality measures, it is uncertain whether they are connected to each other or not, since they may reside in separated parts of the co-investment graph. Even if sometime in the past, the two investors may have had an edge deemed as a co-investment, it is not necessarily true that they made an actual deal to coinvest together. That is, it may have happened by chance that as high degree nodes, they simply invested in the same company within the same year.

Despite the reasonable accuracy of our model, we have reason to suspect that our feature vector can be improved. Fig 8 shows a regression model that attempts to find a linear separation boundary between the two classes, based on minimizing the logistic loss. Immediately, we see a strong multicollinearity between the Weighted Katz and Adamic-Adar metrics. Multicollinearities between features often result in high variances of the coefficients of the features. This is also corroborated by the Variance Inflation Factor (VIF) value of Weighted Katz ($6498.37 >> 5$) on that particular example. This result implies we should only consider one of these features or the other as they relay roughly the same information in our network.

Table II shows a confusion matrix for link prediction on the coinvestment network in 2014. Due to the higher number of false negatives (predicted non-edge, but actually edge) than number of false positives (predicted edge, but actually non-edge), our algorithm seems to have higher precision than recall. However, it may be better in true prediction settings to sacrifice precision for recall since the nature of our prediction is such that it is better to more liberal about our predictions than conservative. However, we can lower precision and increase recall, in the logistic model by decreasing the threshold for the logistic score (by default, the score threshold is 0 which denotes 50% probability labeled as positive) for positive prediction.

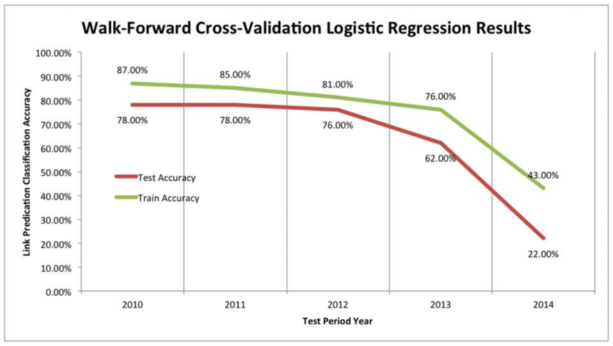


Fig. 7: Results of running Walk Forward Cross-Validation with a window of $T = 2$ years on test periods from 2009 to 2014. The model used was Logistic Regression with both the Adamic Adar and Weighted Katz features. The mean test accuracy from this walk-forward cross-validation is 63.20%. However, as we see, our accuracy deteriorates over time, indicating that our model performance may not be robust to variations in time.

TABLE II: Confusion Matrix for Link Prediction on 2014 Folded Coinvestment Network F (Train Period 2013)

		Predicted		
		Non-Edge	**Edge**	**Total**
Actual	**Non-Edge**	624	61	685
	Edge	244	152	396
	Total	868	213	

Using our model to predict links in the coinvestment network, we have a 2x2 confusion matrix on the predicted vs. actual labels of the test set. We see that our predictions skew disproportionately toward predicting edges, having more false positives than false negatives.

B. Bipartite Link Prediction

We consider our model to predict investments from investors to companies. Unfortunately as we show in Fig 7, we can see that our model shows a significant decrease in accuracy in walk-forward cross validation as compared to accuracies of the coinvestment link prediction model. We also can see from the figure that the accuracy deteriorates over time, perhaps because the same model may not be robust to variations in time. Patterns in investor-company investments may fundamentally change over time. From observational data, we saw that in 2014, there were significantly more investments in years prior, and our model perhaps did not account for this increase, and was too conservative in predicting true edges (leading to more false negatives) in the face of a changing global venture capital system.

Table III shows a 2x2 confusion matrix for link prediction on the bipartite network in 2014. It has a corresponding total accuracy of 72.0%, again significantly lower than the coinvestment problem. While the accuracy score is decent, this does not necessarily mean that that we can predict an edge between a specific investor and a specific company; this is because we severely down-sampled the number of non-edges in our training and test set so that the prediction task would be feasible. This automatically introduces bias into our model toward being able to predict edges correctly since the ratio of edges to non/edges is now half instead of in reality being a much smaller fraction. When we remove the down-sampling, we discover the true reality of the difficulty of link prediction - our model predicts all non-edges in this scenario and our features hardly add any value.

IX. CONCLUSION

Despite having a very rich and complex dataset, we've found that the link prediction problem is incredibly difficult in investment networks for several reasons. First is informational asymmetry and the nature of venture capital investments: investments are made by venture capitalists who spend a significant effort in *qualitatively* vetting a company's team and product; this is information that is not able to be captured in simple and surface level features such as where the company is located, how much they have previously raised, and which market segment they exist in. Second is the highly dynamic nature and constant variance in terms of how investments occur; as we profiled the data and examined the network structure at different points in time, we noticed that these were highly dynamic and constantly changing. The results from our link prediction problem on the bipartite investment graph exactly illustrate this sentiment. To relax the problem, a looser prediction problem was considered based on the folded coinvestment network, and predicting if

TABLE III: Confusion Matrix for Link Prediction on 2014 Bipartite Investment Network G (Train Period 2013)

		Predicted		
		Non-Edge	**Edge**	**Total**
Actual	**Non-Edge**	331	225	556
	Edge	86	470	556
	Total	417	695	

Using Adamic-Adar as our feature for predicting investment links in our bipartite investment graph, we note that the Logistic Regression model does a decent job in terms of accuracy. However, this does not necessarily mean the model is well-suited for an actual investment prediction task since we down-sampled the number of non-edges to a 50%-50% edge / non-edge breakdown to make this prediction task feasible. Note that had we not down-sampled, our model would simply predict all non-edges, since many more non-edges occur than edges in the actual investment network .

an edge would exist in this network, indicating that two investors would investment in the same company, though not specifying which company. This proved to be a more feasible problem given our dataset, since we were able to examine graph properties such as degree distribution which gave us insight into *how often* investors invest and in *which type* of companies they invest in. This allowed us to construct provides with better predictive power. Ultimately, this research has granted us a greater acuity into the difficulties associated with link prediction in graphs, and also the basic toolkit to understanding common features such as Adamic-Adar, Weighted-Katz, and node centrality measures that give us an entry way into exploring the complexities of link prediction problems.

X. ACKNOWLEDGEMENTS

We thank Leon Yao for his helpful comments and insights that allowed us to better scope our project focus.

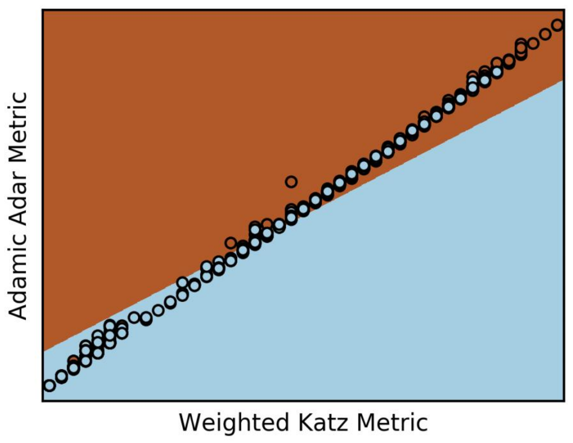


Fig. 8: Results of regression model on link prediction in subgraph of folded coinvestment network F on the top 50 investors in terms of investments made, trained on years 2005 to 2013 and tested on 2014. The line that separates the brown from blue is the classification line and the points plotted are the training sample points and their classification.

XI. INDIVIDUAL CONTRIBUTIONS

Kevin: bipartite link prediction algorithm formulation and coding, data analysis and profiling, constructing data pipeline in Python/Pandas, regression modeling and walk-forward cross-validation implementation (SciKit/Python and R), code for utility files, graph visualizations in SNAP, generating plots, error analysis and explanation, tabulation of final results, final report writeup

Timothy: data cleanup; collaborative link prediction algorithm; data profiling; implementation of graph formulation; generation of predictive features of Weighted Katz and Adamic Adar in Python; construction of edge weights using funding amounts in R; construction of data pipeline in R; error analysis and explanation; final report writeup

Anthony: data analysis and profiling; implementation for Bonacich Centrality; figures and formatting for final report; creating poster

REFERENCES

[1] Bellavitis, Cristiano, Igor Filatotchev, and Vangelis Souitaris. ”The Impact of Investment Networks on Venture Capital Firm Performance: A Contingency Framework.” Brit J Manage British Journal of Management (2016): n. pag. Web.

[2] Liang, Yuxian Eugene, and Soe-Tsyr Daphne Yuan. ”Predicting Investor Funding Behavior Using Crunchbase Social Network Features.” Internet Research 26.1 (2016): 74-100. Web.

[3] Liben-Nowell, David, and Jon Kleinberg. ”The Link Prediction Problem for Social Networks.” Proceedings of the Twelfth International Conference on Information and Knowledge Management - CIKM ’03 (2003): n. pag. Web.

[4] Jean. ”The Crunchbase Graph : importing data into Neo4j”. Linkurious. Web. ”https://linkurio.us/crunchbase-graph-mporting-data-neo4j

[5] Zhang, Charles, Chan, Ethan, and Abdulhamid, Adam. ”Link Prediction in Bipartite Venture Capital Investment Network”. CS 224W Report. Stanford University. Web.